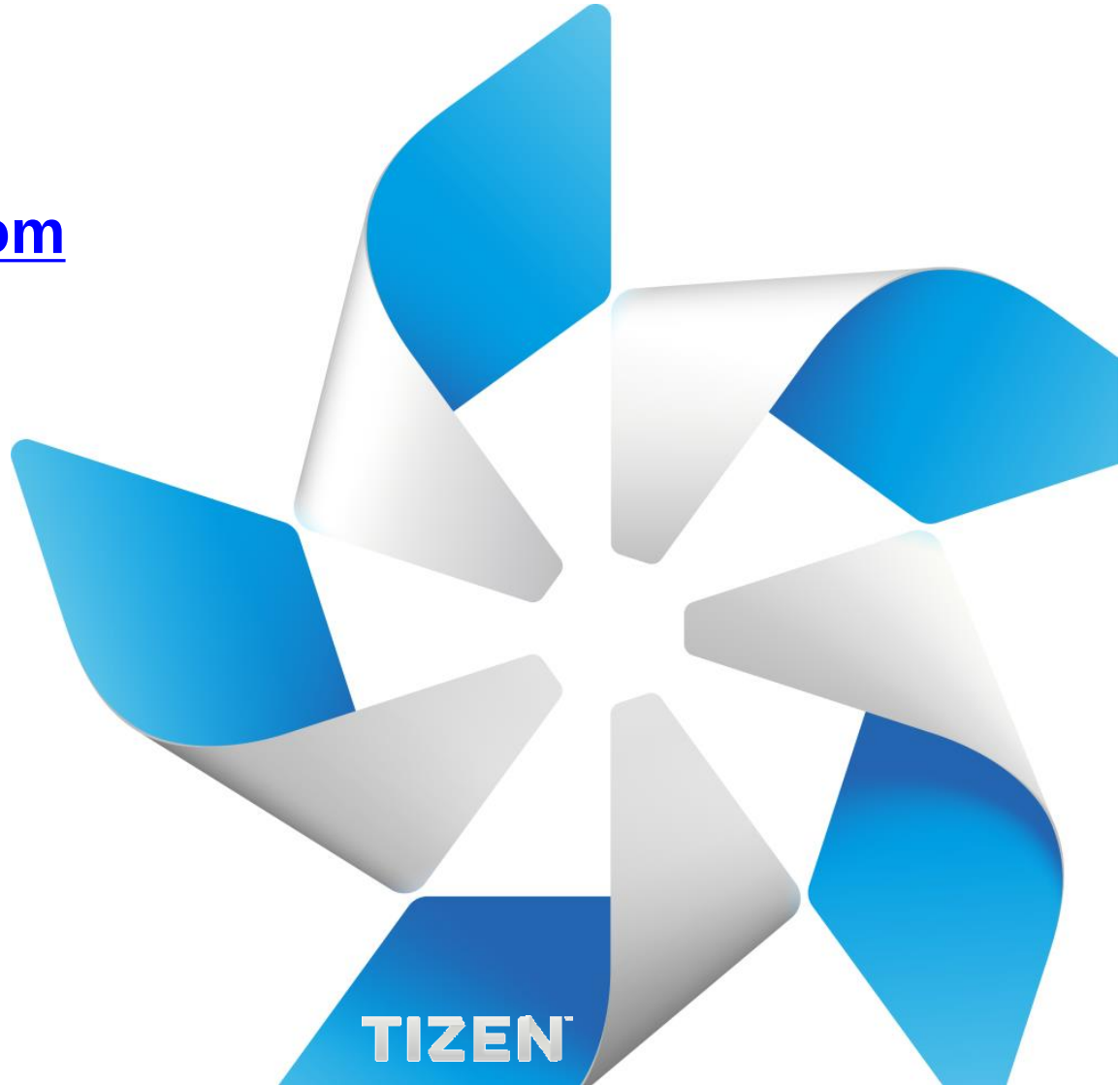
How to develop the ARM 64bit board, Samsung TM2 with Exynos5433

Chanwoo Choi cw00.choi@samsung.com

Seung-Woo Kim sw0312.kim@samsung.com

SW Center,

Samsung Electronics



Who are we?

Chanwoo Choi

Kernel Maintainer of EXTCON and Exynos Clock.
Maintainer of Tizen Kernel.

<cw00.choi@samsung.com>

Seung-Woo Kim

Kernel Maintainer of Exynos DRM.
Maintainer of Tizen Kernel.

<sw0312.kim@samsung.com>

- What is Samsung TM2?
- What we do for ARM 64bit?
- What are difficult issues on TM2?

What is Samsung TM2?

Samsung TM2 (Tizen Mobile)

- TM2 is the mobile reference board for Tizen.
- Uses the Samsung Exynos5433 SoC.
- ARMv8 architecture and supporting 64bit.

Development history of TM2

- Tizen starts supporting 64bit on ARM Juno board.
- TM2 board development started on Oct. 2014, based on Exynos5433 announced on Sep. 2014.
- Now, Tizen supports almost all functionalities except for modem.
- First patch posted on mainline on Nov. 2014.

Samsung TM2 Kernel information

- Tizen git repository with v4.1 kernel
  - More than 1,300 patches for Samsung TM2 based on Exynos5433.
  - https://review.tizen.org/git/?p=platform/kernel/linux-exynos.git;a=shortlog;h=refs/heads/tizen

projects / platform / kernel / linux-exynos.git / shortlog

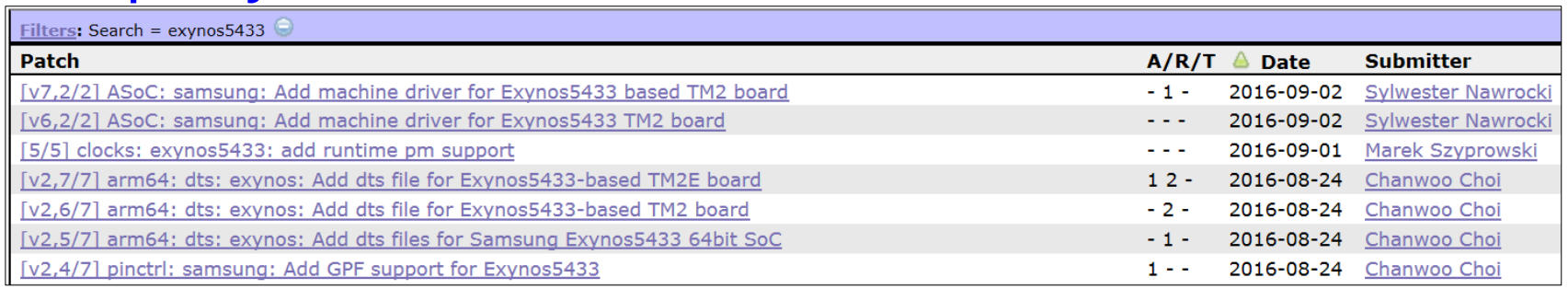
summary | shortlog | log | commit | commitdiff | tree
first · prev · next

platform/kernel/linux-exynos.git

12 hours ago	Andrzej Hajda	drm/panel/s6e3ha2: remove redundant definition of VINT_... 56/87156/4 tizen
12 hours ago	Andrzej Hajda	drm/panel/s6e3ha2: fix includes 55/87155/4
12 hours ago	h.sandeep	Bluetooth: Add MGMT tizen_handlers and TIZEN_OP_BASE_CODE. 23/86423/4
13 hours ago	Sudha Bheemanna	Bluetooth: Add "TIZEN_BT" flag 72/85072/8
11 days ago	Seung-Woo Kim	drm/exynos/hdmi: workaround to check invalid modes... 87/87187/3 accepted/tizen_common accepted/tizen_ivi accepted/tizen/tv/20160912.093425 accepted/tizen/wearable/20160912.093442 submit/tizen/20160912.021917
11 days ago	Sylwester Nawrocki	Merge "Enable dummy_hcd and FunctionFS in tm2 config...
12 days ago	Hoegeun Kwon	arm64: dts: exynos5433: add sleep state for BCM4773... 15/87015/4

Patches on Linux Kernel mailing list

- More than 250 patches for TM2 and Exynos5433 including old versions.
  - https://patchwork.kernel.org/project/linux-samsung-soc/list/?q=exynos5433&state=*&archive=both



Filters: Search = exynos5433

Patch	A/R/T	Date	Submitter
[v7,2/2] ASoC: samsung: Add machine driver for Exynos5433 based TM2 board	- 1 -	2016-09-02	Sylwester Nawrocki
[v6,2/2] ASoC: samsung: Add machine driver for Exynos5433 TM2 board	- - -	2016-09-02	Sylwester Nawrocki
[5/5] clocks: exynos5433: add runtime pm support	- - -	2016-09-01	Marek Szyprowski
[v2,7/7] arm64: dts: exynos: Add dts file for Exynos5433-based TM2E board	1 2 -	2016-08-24	Chanwoo Choi
[v2,6/7] arm64: dts: exynos: Add dts file for Exynos5433-based TM2 board	- 2 -	2016-08-24	Chanwoo Choi
[v2,5/7] arm64: dts: exynos: Add dts files for Samsung Exynos5433 64bit SoC	- 1 -	2016-08-24	Chanwoo Choi
[v2,4/7] pinctrl: samsung: Add GPF support for Exynos5433	1 - -	2016-08-24	Chanwoo Choi

  - https://patchwork.kernel.org/project/linux-samsung-soc/list/?q=TM2&state=*&archive=both
- More than 100 patches for TM2 and Exynos5433 have been merged.

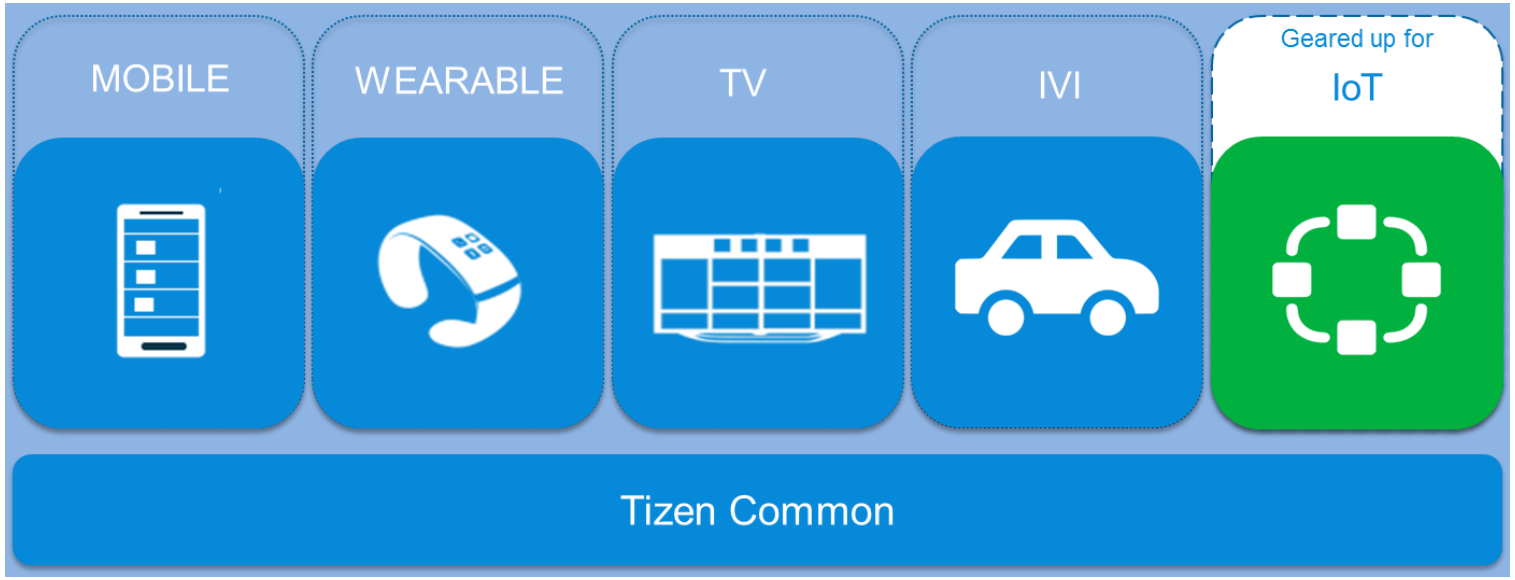
Samsung Exynos5433

- Exynos5433 appeared on Sep. 2014.

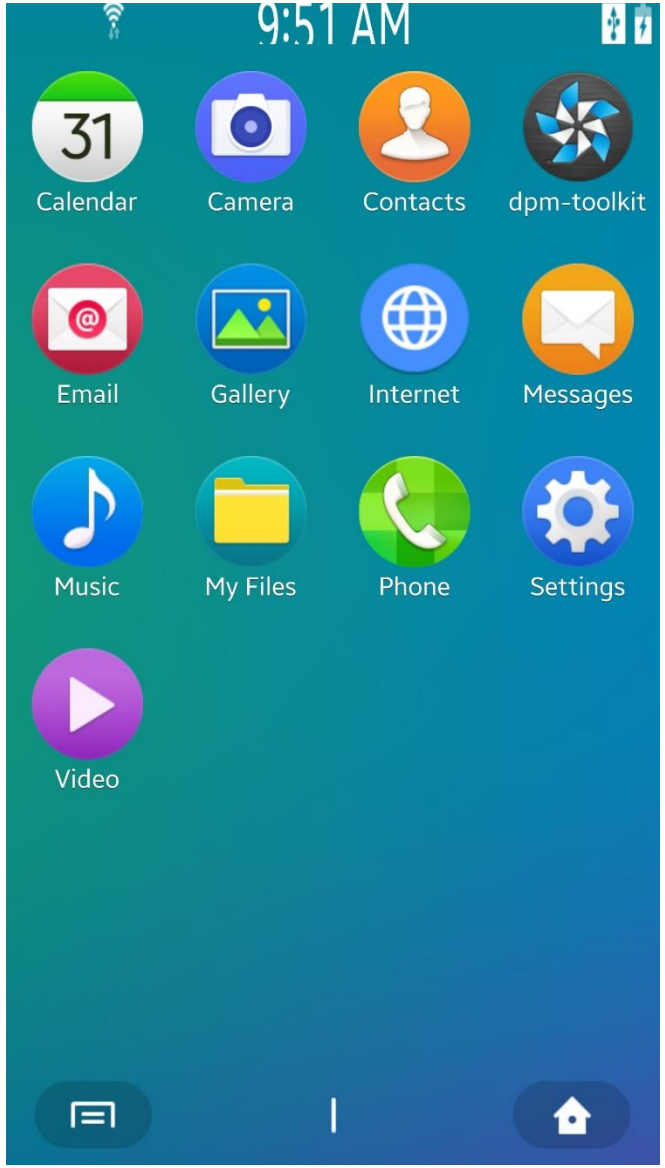
Module	Description
CPU	Octal Core CPU - Quad Cortex-A57 1.9GHz + Quad Cortex-A53 1.3GHz
Memomy	Memory bandwidth 1066MHz
Display	WQXGA (2560 x 1600) or WQHD (2560 x 1440)
Video	1080p 120-frame and UHD 30-frame
GPU	3D graphics hardware
High speed interface	PCIe (Peripheral Component Interconnect Express) LLI (Low Latency Interface) eMMC 5.1 USB 3.0

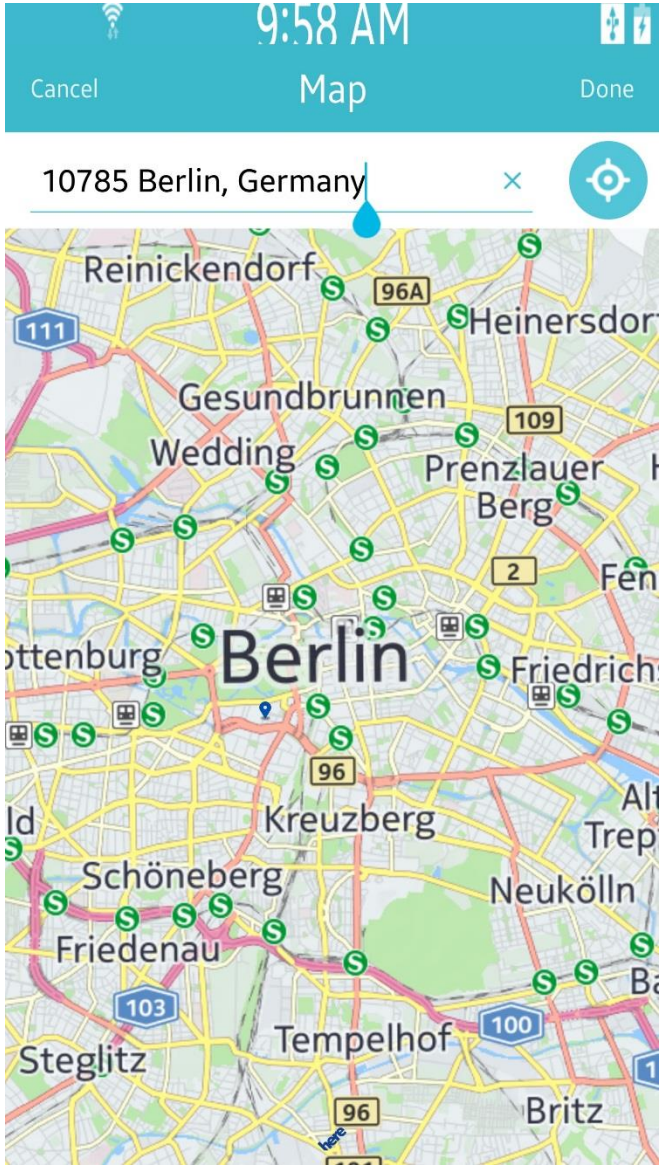
What is Tizen ?

www.tizen.org



Tizen on TM2


9:51 AM
Calendar
Camera
Contacts
dpm-toolkit
Email
Gallery
Internet
Messages
Music
My Files
Phone
Settings
Video


9:58 AM
Cancel
Map
Done
10785 Berlin, Germany
Reinickendorf
96A
111
Heinersdor
Gesundbrunnen
109
Wedding
Prenzlauer
Berg
2
Berlin
Friedrich
96
Kreuzberg
Schöneberg
Neukölln
Friedenau
103
100
Tempelhof
Steglitz
96
Britz

CPU Bechmark result on TM2

- With Dhrystone, ARM 64bit shows better performance than ARM 32bit.
  - Dhrystone : 20% improvement on ARM 64bit

Benchmark Tools	Status	ARM 64bit	ARM 32bit
Dhrystone	DMIPS per second	12626582.4	10549179.0

What we do for ARM 64bit?

for 64bit

for CPU cores

ARMv8

- ARMv8 architecture support the 64bit.

- What are the difference between 32bit and 64bit?
  - Support the 64bit memory map
  - PSCI (Power State Coordinate Interface)

What we do for ARM 64bit?

for 64bit

for CPU cores

64bit memory address mapping

- The bootloader should pass the physical memory address through ATAG_MEM parameter with 64bit address.
  - ‘mem=<size>[KM][,@<phys_offset>]

- Memory node in TM2

```
/*
 * start address          : 0x00000000_20000000
 * memory bank size  : 0x00000000_c0000000
 */
memory@20000000 {
        device_type = "memory";
        reg = <0x0 0x20000000 0x0 0xc0000000>;
};
```

The exynos5433-tm2.dtsi include the memory Device-Tree node.

ARM drivers for 64bit : case of exynos-iommu

- SYSMMU v5 of exynos5433
  - Supporting 36bit physical address space
  - Shifting all page entry values by 4bits
  - iommu: exynos: add support for v5 SYSMMU (merged)

ARM drivers for 64bit : case of s5p-mfc

- No 64bit considered address/offset size
  - media: s5p-mfc: fix mmap support for 64bit arch (merged)
    - Fix offset base depends on the systems architecture
  - media: s5p-mfc: fix broken pointer cast on 64bit arch (merged)
    - Remove pointer casting with fixed 32bit size

ARM drivers for 64bit : case of dw_mmc exynos

- AArch64 gcc bug in right-shift

```
s8 i;
u8 c, __c;

c = 0x7f;

for (i = 0; i < 8; i++) {
    __c = ror8(c, i);
}
```

```
# expecetd

i == 0 → __c == 0x7f
i == 1 → __c == 0xbf
i == 2 → __c == 0xdf
i == 3 → __c == 0xef
i == 4 → __c == 0xf7
i == 5 → __c == 0xfb
i == 6 → __c == 0xfd
i == 7 → __c == 0xfe
```

```
# result

i == 0 → __c == 0x7f 0111_1111
i == 1 → __c == 0x3f 0011_1111
i == 2 → __c == 0x1f 0001_1111
i == 3 → __c == 0x0f 0000_1111
i == 4 → __c == 0x07 0000_0111
i == 5 → __c == 0x03 0000_0011
i == 6 → __c == 0x01 0000_0001
i == 7 → __c == 0x00 0000_0000
```

- Workaround from kernel in early stage

- Fixed with GCC patch

Runtime checker: KASAN

- Kernel Address SANitizer
  - For ARM 64bit, since v4.3
  - CONFIG_KASAN

  - Can detect out-of-bounds accesses and use-after-free bugs.

KASAN – case of extcon-max77843

- extcon: max77843: add guard element in array (fixed from tizen devel tree)
  - https://review.tizen.org/git/?p=platform/kernel/linux-exynos.git;a=commitdiff;h=5ec7e9f3010a9254b89c3307385a8bda658d37cb

```
BUG: KASan: out of bounds access in extcon_dev_register+0xc0/0x978 at
addr ffffffc001bcc440
page dumped because: kasan: bad access detected
Address belongs to variable max77843_extcon_cable+0x60/0xf00
Call trace:
[...]
[<ffffffc00021e9ec>] kasan_report+0x44/0x50
[<ffffffc00021d2dc>] __asan_load8+0x94/0xb0
[<ffffffc000a78904>] extcon_dev_register+0xbc/0x978
[<ffffffc000a791f8>] devm_extcon_dev_register+0x38/0x90
[<ffffffc000a7cab8>] max77843_muic_probe+0x1e0/0x5f0
[...]
Memory state around the buggy address:
 ffffffc001bcc300: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 ffffffc001bcc380: 00 00 00 00 00 00 00 00 fa fa fa fa 00 00 00 00
>ffffffc001bcc400: 00 00 00 00 00 00 00 00 fa fa fa fa 00 00 00 00
                                  ^
 ffffffc001bcc480: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 ffffffc001bcc500: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
@@ -149,6 +149,7 @@ static const char *max77843_extcon_cable[] = {
...
+       NULL,
 };
```

KASAN – case of fimc-is

- fimc-is: fix wrong index access for dt child nodes (fixed from tizen devel tree)
  - https://review.tizen.org/git/?p=platform/kernel/linux-exynos.git;a=commit;h=f612545917b4b303d115655c7e4e5814b4969f15

```
BUG: KASan: use after free in fimc_is_parse_children_dt+0x6c/0xe8 at
addr ffffffc08d27ffa8
page dumped because: kasan: bad access detected
Call trace:
[...]
[<ffffffc00021e9ec>] kasan_report+0x44/0x50
[<ffffffc00021d38c>] __asan_store8+0x94/0xb0
[<ffffffc000991900>] fimc_is_parse_children_dt+0x68/0xe8
[<ffffffc000959368>] fimc_is_probe+0xc0/0xed8
[...]
Memory state around the buggy address:
 ffffffc08d27fe80: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
 ffffffc08d27ff00: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
>ffffffc08d27ff80: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
                                  ^
 ffffffc08d280000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 ffffffc08d280080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
...
  i = of_alias_get_id(child, "fimc-lite");
- if (i >= 0 || i < FIMC_IS_MAX_NODES)
+ if (i >= 0 && i < FIMC_IS_MAX_NODES)
          core->lite_np[i] = child;

  i = of_alias_get_id(child, "csis");
- if (i >= 0 || i < FIMC_IS_MAX_NODES)
+ if (i >= 0 && i < FIMC_IS_MAX_NODES)
          core->csis_np[i] = child;
...
```

Runtime checker: UBSAN

- Undefined Behavior SANitizer
  - For ARM 64bit, since v4.5
  - CONFIG_UBSAN / CONFIG_UBSAN_SANITZE_ALL
  - Can detect overflows, shift out-of-bonds, array out-of-bounds and more undefined behaviors.

UBSAN – case of dwmmc

- mmc: dw_mmc: remove UBSAN warning in dw_mci_setup_bus() (merged to mainline)

```
UBSAN: Undefined behaviour in drivers/mmc/host/dw_mmc.c:1102:14
shift exponent 250 is too large for 32-bit type 'unsigned int'
Call trace:
[<ffffff90080908a8>] dump_backtrace+0x0/0x380
[<ffffff9008090c3c>] show_stack+0x14/0x20
[<ffffff90087457b8>] dump_stack+0xe0/0x120
[<ffffff90087b1360>] ubsan_epilogue+0x18/0x68
[<ffffff90087b1a94>] __ubsan_handle_shift_out_of_bounds+0x18c/0x1bc
[<ffffff9008d89cb8>] dw_mci_setup_bus+0x3a0/0x438
[...]

UBSAN: Undefined behaviour in drivers/mmc/host/dw_mmc.c:1132:27
shift exponent 250 is too large for 32-bit type 'unsigned int'
Call trace:
[<ffffff90080908a8>] dump_backtrace+0x0/0x380
[<ffffff9008090c3c>] show_stack+0x14/0x20
[<ffffff90087457b8>] dump_stack+0xe0/0x120
[<ffffff90087b1360>] ubsan_epilogue+0x18/0x68
[<ffffff90087b1a94>] __ubsan_handle_shift_out_of_bounds+0x18c/0x1bc
[<ffffff9008d89c9c>] dw_mci_setup_bus+0x384/0x438
[...]
```

```
-  if ((clock << div) != slot->__clk_old || force_clkinit)
-          dev_info(&slot->mmc->class_dev,
-                   "Bus speed (slot %d) = %dHz (slot req %dHz, actual %dHZ div = %d)₩n",
-                   slot->id, host->bus_hz, clock,
-                   div ? ((host->bus_hz / div) >> 1) :
-                   host->bus_hz, div);
...
- /* keep the clock with reflecting clock dividor */
- slot->__clk_old = clock << div;   # div == 250
```

UBSAN – case of pinctrl-exynos

- pinctrl: samsung: fix wakeup irq for extended eint (fixed from tizen devel tree)
  - https://review.tizen.org/git/?p=platform/kernel/linux-exynos.git;a=commitdiff;h=367c75d2942b5fadf6faf92d06d096deb72de945;hp=ffeae979bfb99e666e7d9801a57eeac9b6962fad

```
UBSAN: Undefined behaviour in drivers/pinctrl/samsung/pinctrl-exynos.c:376:26
shift exponent 8217 is too large for 64-bit type 'long unsigned int'
Call trace:
[<ffffffc00008f440>] dump_backtrace+0x0/0x218
[<ffffffc00008f668>] show_stack+0x10/0x20
[<ffffffc00159f3b8>] dump_stack+0x80/0xfc
[<ffffffc00159f558>] ubsan_epilogue+0x10/0x6c
[<ffffffc00159fe38>] __ubsan_handle_shift_out_of_bounds+0x188/0x1bc
[<ffffffc000785514>] exynos_wkup_irq_set_wake+0x104/0x138
[<ffffffc0001647b0>] set_irq_wake_real+0x70/0xc0
[<ffffffc000164a70>] irq_set_irq_wake+0x158/0x1b0
[...]
```

```
-  unsigned long bit = 1UL << (2 * bank->eint_offset + irqd->hwirq);      # 8217
...
-        EXYNOS5433_PIN_BANK_EINTW_EXT(4, 0x060, "gpf3", 0x100c),
```

What we do for ARM 64bit?

for 64bit

for CPU cores

PSCI (Power State Coordinate Interface)

- Standard interface for power management by ARM.
- The aim of this standard is to ease the integration between supervisory software from different vendors working at different privilege levels.
- Generalization of the code for Power Management.

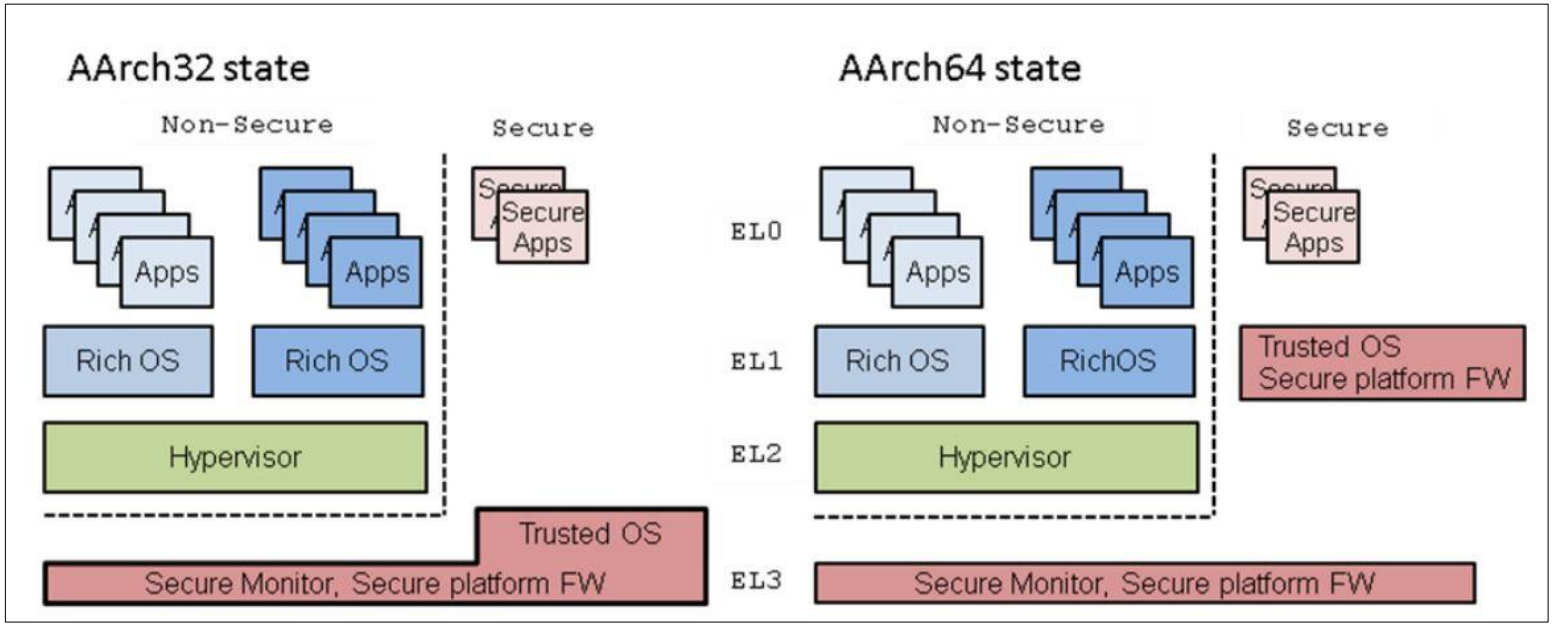
Power management of PSCI

- Power management scenarios:
  - Core idle management (suspend, idle)
  - Core Hotplug
  - System Shutdown and reset.
  - big.LITTLE migration.

- PSCI does not cover DVFS (Dynamic Voltage Frequency Scaling) or device power management (such as GPUs).

Exception Level of PSCI

- Version
  - major number : Define the different interface
  - minor number : Support the compatibility for low version
  - PSCI 0.1 (Samsung Exynos5433)
- ARM Exception Level



Example for suspend with PSCI

- PSCI vs. Legacy method for suspend

```
static int psci_suspend_finisher(unsigned long index)
{
        struct psci_power_state *state = __this_cpu_read(psci_power_state);

        return psci_ops.cpu_suspend(state[index-1], virt_to_phys(cpu_resume));
}
```

in **arch/arm64/kernel/psci.c**

```
static int exynos_suspend(unsigned long resume_addr)
{
        writel(EXYNOS_SLEEP_MAGIC, S5P_VA_SYSRAM_NS + 0xC);
        writel(resume_addr, S5P_VA_SYSRAM_NS + 0x8);
        exynos_smc(SMC_CMD_SLEEP, 0, 0, 0);
        return 0;
}
```

in **arch/arm/mach-exynos/firmware.c**

Example for CPU Hotplug with PSCI

- PSCI vs. Legacy method for CPU Hotplug

```
const struct cpu_operations cpu_psci_ops = {
        .name           = “psci”,
        .cpu_init       = cpu_psci_cpu_init,
        .cpu_prepare    = cpu_psci_cpu_prepare,
        .cpu_boot       = cpu_psci_cpu_boot,
        .cpu_disable    = cpu_psci_cpu_disable,
        .cpu_die        = cpu_psci_cpu_die,
        .cpu_kill       = cpu_psci_cpu_kill,
};
```

in **arch/arm64/kernel/psci.c**

```
struct smp_operation exynos_smp_ops __initdata = {
        .smp_init_cpus          = exynos_smp_init_cpus,
        .smp_prepare_cpus       = exynos_smp_prepare_cpus,
        .smp_secondary_init     = exynos_secondary_init,
        .smp_boot_secondary     = exynos_boot_secondary,
        .cpu_idle               = exynos_cpu_idle,
};
```

in arch/arm/mach-exynos/platsmp.c

How can I support the big.LITTLE core?

- Exynos5433 has the big.LITTLE core.

- Linux kernel includes the IKS (In-Kernel Switcher) from Linaro to support the scheduling for bitg.LITTLE cores.

- But, In-Kernel Switcher does not use the all cores at the same time.
  - To support the high-performance, need to use the all cores simultaneously.

HMP (Heterogeneous Multi Processing)

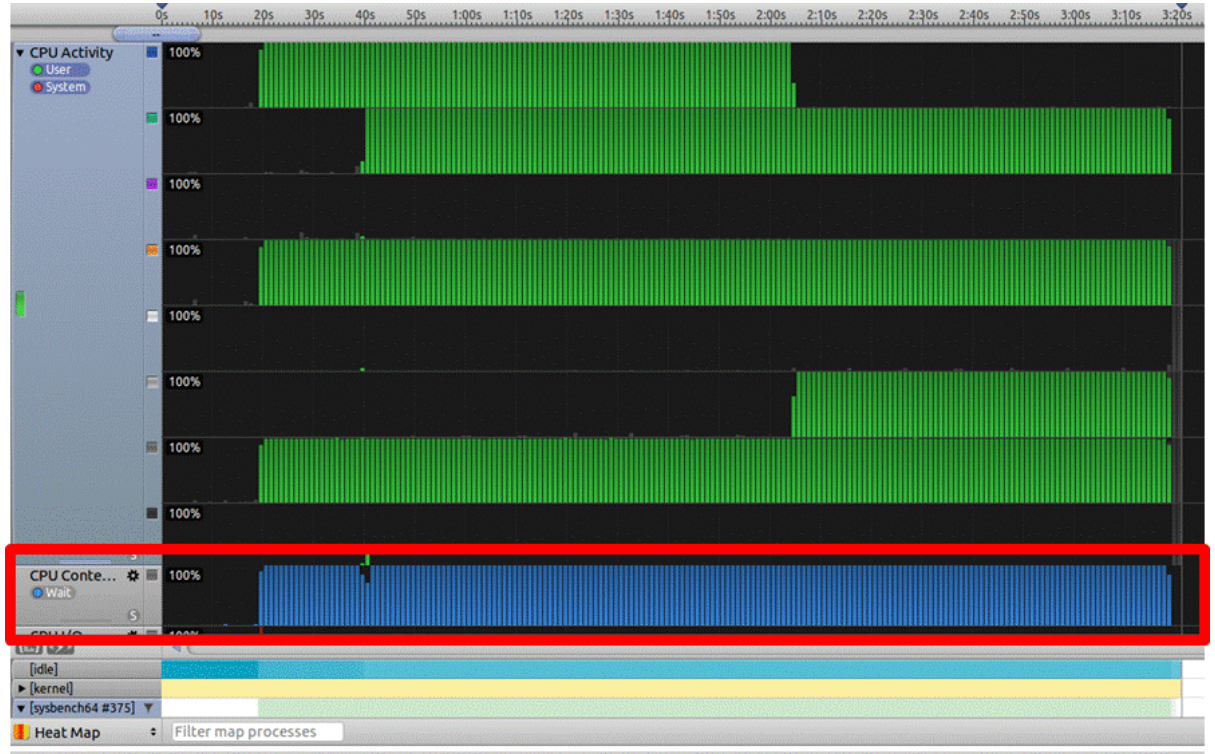
- Linaro provides the HMP to support the big.LITTLE cores scheduling.
- All cores in the system can be used at the same time.
- HMP is supported only on Linux Kernel 3.10
  - Samsung TM2 support the HMP on Linux Kernel v4.1 (LTS version).

HMP - sysbench test

- Take simple benchmark test to check the effect and performance improvement by HMP.
  - benchmark tool : sysbench with 6 threads
    - sysbench64 --test=cpu --num-threads=6 --cpu-max-prime=300000 run
  - profiling tool : ARM DS-5

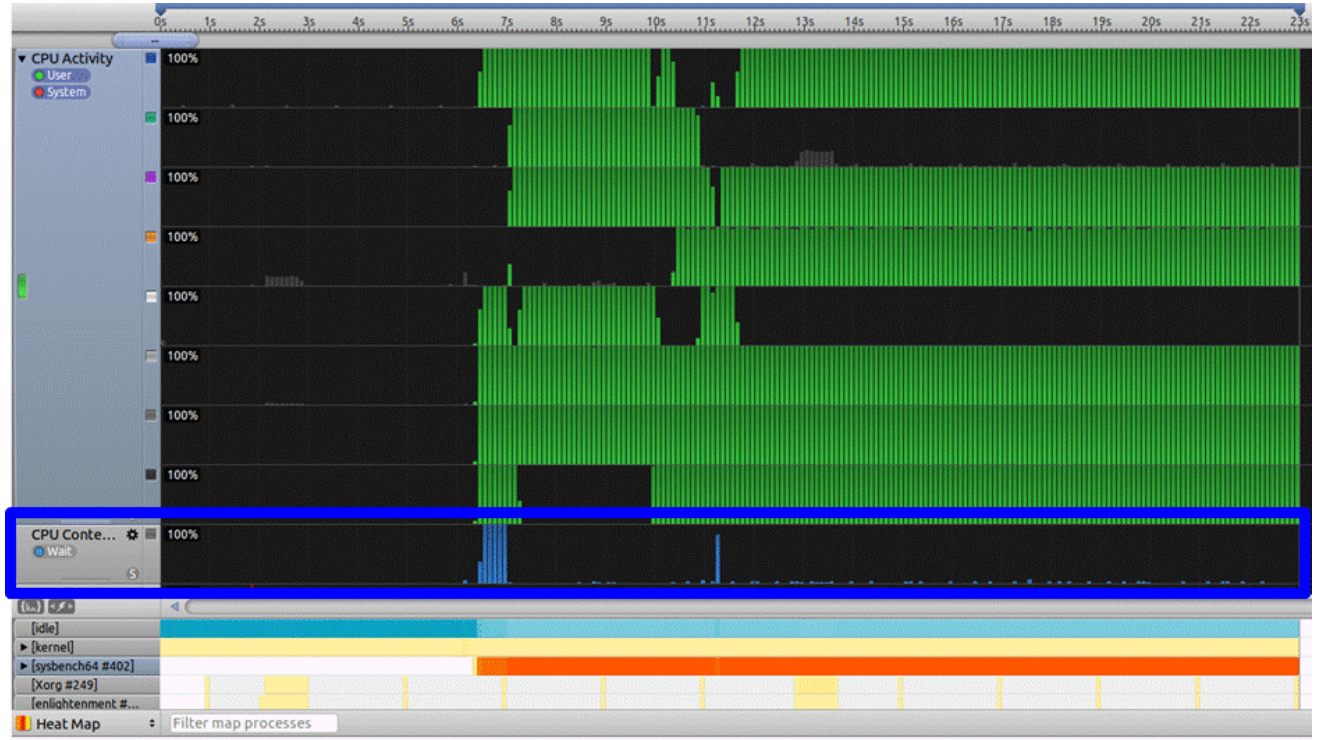
HMP - sysbench test without HMP

- Without HMP, the performance regression happens due to CPU contention.
  - CPU contention can happen if they are too many processes and not enough cores to handle them.
    - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0482h/BABFBJIC.html



HMP - sysbench test with HMP

- If CPU contention does not happen.
  - TM2 can use the whole resources to get high performance on specific scenario such as UHD video playback.



What are the challenges on TM2?

32bit process running on 64bit kernel

Issues required to be discussed on mainline

Personality 32bit setting on 64bit kernel (1/6)

- ARM 64bit kernel provides different information to userland according to the type of running user-process.
  - uname -m
  - cat /proc/cpuinfo

- The user process uses information provided by kernel such as thumb, fp and so on.

Personality 32bit setting on 64bit kernel (2/6)

- Example,
  - The Unity engine on Tizen checks whether thumb mode is supported or not through /proc/cpuinfo.
  - ARM 64bit kernel doesn't include the 'thumb' on the feature /proc/cpuinfo/. But, ARM 64bit supports the 'thumb'.

```
root@localhost:~# cat /proc/cpuinfo
processor       : 0
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32
CPU implementer : 0x41
CPU architecture: 8
.....
```

Personality 32bit setting on 64bit kernel (3/6)

– The 32bit user-process should set the PER_LINUX32 personality type with ‘personality()’ function on runtime.
  • personality(PER_LINUX32);

```
#include <stdio.h>
#include <sys/personality.h>

int main(void)
{
        personality(PER_LINUX32);
        system("cat /proc/cpuinfo");
        system("uname –a");

        return 0;
}
```

Personality 32bit setting on 64bit kernel (4/6)

- ARM 64bit Kernel + 32bit User-process

```
root@localhost:~# uname -m
aarch64

root@localhost:~# cat /proc/cpuinfo
processor       : 0
BogoMIPS        : 48.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32
CPU implementer : 0x41
CPU architecture: 8
......
```

Personality 32bit setting on 64bit kernel (5/6)

- ARM 64bit Kernel + 32bit User-process with PER_LINUX32

```
root@localhost:~# uname -m
armv8l

root@localhost:~# cat /proc/cpuinfo
processor       : 0
model name      : ARMv8 Processor rev 1 (v8l)
BogoMIPS        : 48.00
Features        : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva
                  idivt lpae evtstrm aes pmull sha1 sha2 crc32
CPU implementer : 0x41
CPU architecture: 8
......
```

Personality 32bit setting on 64bit kernel (6/6)

- How to support the compatibilty on 64bit kernel.
  - COMPAT_PSR_*_BIT means the AArch32 CPSR bits to support the compatibility for 32bit user-process.
  - For example for ‘thumb’ mode,
    - There is no CONFIG_THUMB2_KERNEL on ARM 64bit.
    - If COMPAT_PSR_T_BIT is set, the A32/T32 instruction are converted to A64 instruction.

32bit process running on 64bit Kernel

- CONFIG_COMPAT

- Should consider fops->compat_ioctl() if driver has fops->unlocked_ioctl()
  - Usually, not in legacy arm drivers

- Architecture dependent arg type of ioctl
  - Need to consider different size of type from 32bit user

compat_ioctl: case of tgl

- Legacy driver tgl
  - Bad design of legacy driver
  - But need to support same ioctl user interface

- misc: tizen_global_lock: add support for 32bt compat mode from 64bit
  (fixed from tizen devel tree)
  - https://review.tizen.org/git/?p=platform/kernel/linux-exynos.git;a=commitdiff;h=b6516ba567384295bd906e515e31fef3e412fc19;hp=f874e761defc229f415eeb50e66d5cb6000cfb62

compat_ioctl: case of tgl

```
@@ -764,6 +776,9 @@ static const struct file_operations tgl_ops = {
        .open = tgl_open,
        .release = tgl_release,
        .unlocked_ioctl = tgl_ioctl,
+#ifdef CONFIG_COMPAT
+       .compat_ioctl = tgl_ioctl,
+#endif
 };
```

```
...
 #define TGL_IOC_DUMP_LOCKS      _IOW(TGL_IOC_BASE, TGL_DUMP_LOCKS, void *)
+#define TGL_IOC_DUMP_LOCKS_COMPAT                                      ₩
+                                _IOW(TGL_IOC_BASE, TGL_DUMP_LOCKS, unsigned int)
 ...

@@ -649,6 +649,9 @@ static long tgl_ioctl(struct file *file, unsigned int cmd, unsigned long arg)

        switch (cmd) {
        case TGL_IOC_INIT_LOCK:
+#ifdef CONFIG_COMPAT
+       case TGL_IOC_INIT_LOCK_COMPAT:
+#endif
                /* destroy lock with attribute */
                err = tgl_init_lock(session_data, (struct tgl_attribute *)arg);
                break;
```

32bit process running on 64bit Kernel

- Alignment PAD in arg of ioctl
  - v4l2-compat-ioctl32: fix alignment for ARM64 (merged to mainline)
    - Alignment/padding rules on arm64 and x86 differs

$ cat arch/arm64/include/asm/compat.h
…
typedef s64 compat_s64;
…
	$ cat arch/x86/include/asm/compat.h
	…
	typedef s64 __attribute__((aligned(4))) compat_s64;
	…

32bit process running on 64bit Kernel

- Case of mmap() failiure
  - __USE_FILE_OFFSET64 to call __mmap64 from userspace

What are the challenges on TM2?

32bit process running on 64bit kernel

Issues required to discuss on mainline

FPSIMD seg. fault on suspend-to-ram (1/8)

- The Segmentation fault happened after wake up from the suspended state.
  - 32bit user-process such as bash, udevd and so on.

- The fault only happened on 64bit kernel + 32bit user-process.
  - If both kernel and user-process are 64-bit, the crash did not happened.

FPSIMD seg. fault on suspend-to-ram (2/8)

- But when removing the ‘-O2’ option when buidling the user process, the issue did not occur.

- With ‘-O2’ option, the toolchain uses the ‘FMOV’ instead of ‘MOV’ command.
  - ‘-O2’ is the optimization option for GCC.

FPSIMD seg. fault on suspend-to-ram (3/8)

- Finally, when using the ‘FMOV’ assembly command, the segmentation fault occurred.
  - ‘FMOV’ command is used for FPSIMD on ARM 64bit.
  - ‘FMOV’ is floating-point move to or from general-purpose register without conversion.
    - FMOV Dd, Xn   /* 64-bit to double-precision */
    - FMOV Xn, Dd   /* Double-precision to 64-bit*/
      - ‘Xn’ is the 64-bit name of the general-purpose source register.
      - ‘Dd’ is the 64-bit name of the SIMD and FP destination register.

FPSIMD seg. fault on suspend-to-ram (4/8)

- Result of ‘Dn’ register dump between normal and fail case as following:
  - On fail case, ‘Dn’ registers have garbage data instead of zero or .

FPSIMD seg. fault on suspend-to-ram (5/8)

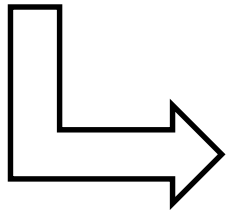
- Register dump between success and fail

<Success case for suspend-to-ram>	<Fail case for suspend-to-ram>
d7 {f = 0x0, u = 0x8020080280200802, s = 0x8020080280200802} {f = -4.4588500238274385e-308, u = 9232388042942056450, s = -9214356030767495166}	d7 {f = 0x7fffffffffffffff, u = 0x7e2c5ba5727e0e15, s = 0x7e2c5ba5727e0e15} {f = 5.9347331789315386e+299, u = 9091742513902784021, s = 9091742513902784021}
d8 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d8 {f = 0x7fffffffffffffff, u = 0x4fdcbf57f44dee9d, s = 0x4fdcbf57f44dee9d} {f = 5.2011338352281976e+76, u = 5754684808354459293, s = 5754684808354459293}
d9 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d9 {f = 0x8000000000000000, u = 0xf5eb59012cb4849a, s = 0xf5eb59012cb4849a} {f = -1.0512031953185022e+260, u = 17720355020399215770, s = -726389053310335846}
d10 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d10 {f = 0x0, u = 0x531a04477effd23, s = 0x531a04477effd23} {f = 1.1853291409126618e-283, u = 374256459978898723, s = 374256459978898723}
d11 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d11 {f = 0x0, u = 0x1272f9b865387e26, s = 0x1272f9b865387e26} {f = 8.3991572712913294e-220, u = 1329399410395217446, s = 1329399410395217446}
d12 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d12 {f = 0x0, u = 0x385e0dfcecfd4fd6, s = 0x385e0dfcecfd4fd6} {f = 3.5329060230148928e-37, u = 4061699293893709782, s = 4061699293893709782}
d13 {f = 0x0, u = 0x0, s = 0x0} {f = 0, u = 0, s = 0}	d13 {f = 0x0, u = 0x148d2d6cf610121c, s = 0x148d2d6cf610121c} {f = 1.1093798491827732e-209, u = 1480889798482727452, s = 1480889798482727452}

FPSIMD seg. fault on suspend-to-ram (6/8)

- ARM 64bit kernel handles the FPSIMD registers when context switching as following:

```
/* Thread switching */
struct task_struct __switch_to(
      struct task_struct *prev,  struct task_struct *next)
{
        fpsimd_thread_switch(next);
        ......
}
in arch/arm64/kernel/process.c
```



```
void fpsimd_thread_switch(struct task_struct *next)
{
    ......

    struct fpsimd_state *st = &next->thrad.fpsimd_state;

    if (__this_cpu_read(fpsimd_last_state) == st
            && st->cpu == smp_processor_id())
            clear_ti_thread_flag(task_thread_info(next),
                                 TIF_FOREIGN_FPSTATE);
    else
             set_ti_thread_flag(task_thread_info(next),
                                TIF_FOREIGN_FPSTATE);
    ......
}
in arch/arm64/kernel/fpsimd.c
```

FPSIMD seg. fault on suspend-to-ram (7/8)

- The fpsimd_thread_switch() doesn’t consider the specific situation when CPU wakes up from the suspended state.
  - After wake up from the suspened state, the fpsimd_thread_switch() has to restore the FPSIMD registers always to prevent the segmentation fault.

FPSIMD seg. fault on suspend-to-ram (8/8)

- How to fix the segmentation fault.
  - The fpsimd_thread_switch() should handle the suspend-to-ram for 32bit user-process.
  - Or TIF_FOREIGN_FPSTATE should be always set for next task temporarily as following:

```
void fpsimd_thread_switch(struct task_struct *next)
{
    ......
    set_ti_thread_flag(task_thread_info(next), TIF_FOREIGN_FPSTATE);
    ......
}
in arch/arm64/kernel/fpsimd.c
```

ARM Generic Timer issue in ARM 64bit (1/4)

- Linux kernel needs the clock sources for timekeeping features.
  - clocksource and sched_clock()

- The purpose of the clocksource is to provide a timeline for the system that tells you where you are in time.

- sched_clock() is used for scheduling and timestamping of printk.

ARM Generic Timer issue in ARM 64bit (2/4)

- ARM arch. provides the Generic Timer for system counter such as clocksource and sched_clock().
  - physical counter (CNTPCT_EL0 register in AArch64)
  - virtual counter (CNTVCT_EL0 register in AArch64)

- ARM 64bit kernel and user space requires the use of virtual counter.
  - ARM 64bit only expose virtual counter(CNTVCT) to user VDSO (Virtual Dynamically Linked Shared Object).

ARM Generic Timer issue in ARM 64bit (3/4)

- Exynos5433 fails to synchronize the timestamping among CPUs when printing the log with virtual counter as following:

```
[     0.050577] Mount-cache hash table entries: 8192 (order: 4, 65536 bytes)
[     0.057208] Mountpoint-cache hash table entries: 8192 (order: 4, 65536 bytes)
[     0.065307] ASID allocator initialised with 65536 entries
[     0.094297] EFI services will not be available.
[  244.881736] Detected VIPT I-cache on CPU1
[  244.881801] CPU1: Booted secondary processor [410fd031]
[ 5881.039842] Detected VIPT I-cache on CPU2
[ 5881.039887] CPU2: Booted secondary processor [410fd031]
[ 6253.690734] Detected VIPT I-cache on CPU3
[ 6253.690776] CPU3: Booted secondary processor [410fd031]
[   444.293771] Detected PIPT I-cache on CPU4
[   444.293775] CPU features: enabling workaround for ARM erratum 832075
[   444.293812] CPU4: Booted secondary processor [411fd070]
[   444.309757] Detected PIPT I-cache on CPU5
[   444.309782] CPU5: Booted secondary processor [411fd070]
[   444.325756] Detected PIPT I-cache on CPU6
[   444.325781] CPU6: Booted secondary processor [411fd070]
[   444.341765] Detected PIPT I-cache on CPU7
[   444.341790] CPU7: Booted secondary processor [411fd070]
[     0.211925] Brought up 8 CPUs
```

ARM Generic Timer issue in ARM 64bit (4/4)

- How to fix the issue about random offset on ARM 64bit.
  - Initialize the offset (CNTOFF) to use the Generic Timer on bootloader.
    - Generic Timer comes up with an uninitialized offset (CNTOFF) between virtual and physical counters.
    - Each core gets a different random offset.
  - Use the physical counter if VDSO is not used.

Deadlock between regmap and CCF (1/5)

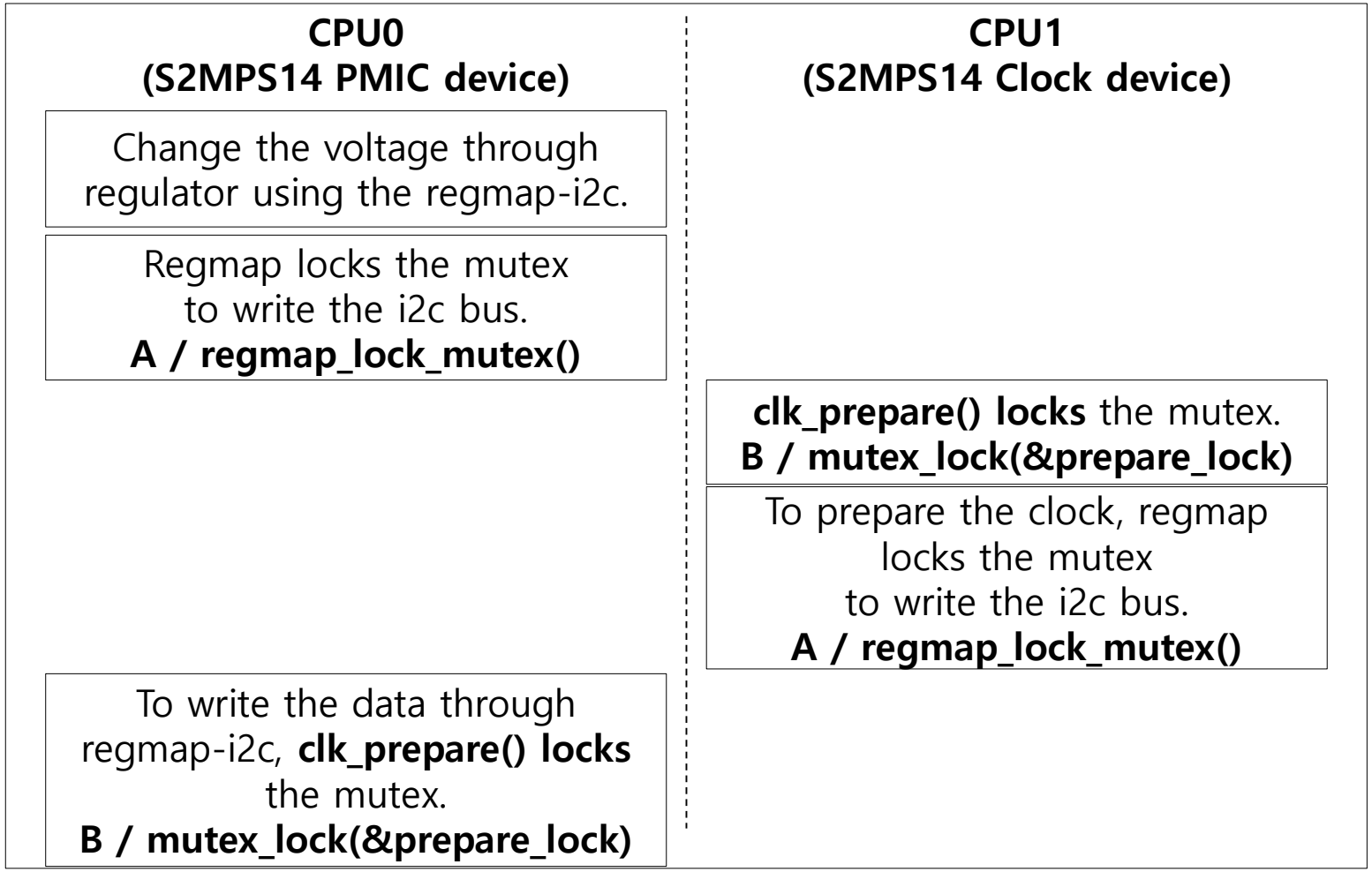
- ABBA deadlock happened between Regmap and CCF (Common Clock Framework).

- Mutex for Regmap vs. Mutex for clk_prepare().

- Example,
  - the MFD device includes the both PMIC and clock device such as Samsung S2MPS1x series.

Deadlock between regmap and CCF (2/5)

- Example of ABBA deadlock



< ABBA deadlock scenario on TM2 with s2mps14>

Deadlock between regmap and CCF (3/5)

- Fix the deadlock between regmap and CCF
  - clk_prepare() should be used on probe and then clk_enable() will be used on runtime temporarily.
  - clk_enable() uses the spinlock instead of mutex.
  - For example,
    - i2c: s3c2410: fix ABBA deadlock by keeping clock prepared
      - https://patchwork.kernel.org/patch/5659351/
    - i2c: exynos5: Fix possible ABBA deadlock by keeping I2C clock prepared
      - https://patchwork.kernel.org/patch/8859551/

Deadlock between regmap and CCF (4/5)

- Fix the deadlock between regmap and CCF
  - Use the spinlock on the regmap-i2c temporarily.
    - regmap-mmio uses the ‘spinlock’ for locking mechanism.
    - regmap-i2c uses the ’mutex’ for locking mechanism.
    - regmap-spi uses the ‘mutex’ for locking mechanism.

```
struct regmap_bus regmap_mmio = {
        .fast_io = true,
};
in drivers/base/regmap/regmap-mmio.c

struct regmap_bus regmap_i2c = {
        .fast_io = false,
};
in drivers/base/regmap/regmap-i2c.c

struct regmap_bus regmap_spi = {
        .fast_io = false,
};
in drivers/base/regmap/regmap-spi.c
```

```
if (bus->fast_io) {
        spin_lock_init(&map->spinlock);
        map->lock = regmap_lock_spinlock;
        map->unlock = regmap_unlock_spinlock;
} else {
        mutex_init(&map->mutex);
        map->lock = regmap_lock_mutex;
        map->unlock = regmap_unlock_mutex;
}
in drivers/base/regmap/regmap.c
```

Deadlock between regmap and CCF (5/5)

- How to fix the deadlock.
  - It is not a fundamental solution to use the spinlock instead of mutex.
  - The locking idea to fix the deadlock was suggested on mainline with RFC patches.
    - [RFC 00/17] clk: Add per-controller locks to fix deadlocks
      - https://lkml.org/lkml/2016/8/16/442

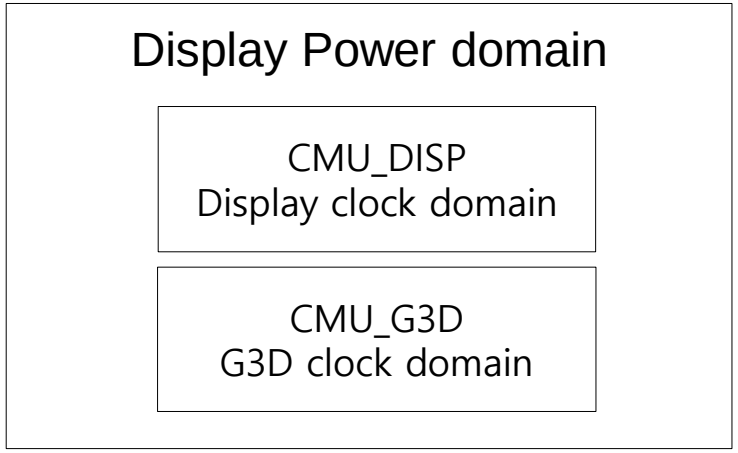
Dependency between power domain and CCF (1/6)

- The graphics h/w modules do not work after wakes up from the suspended state.

- The graphics-related clocks are initialized after wakes up from the suspended state.
  - To save and restore the clock registers on CCF (Common Clock Framework) for the suspend-to-RAM.

Dependency between power domain and CCF (2/6)

- In exynos5433, the clock domain belongs to the power domain about display modules.

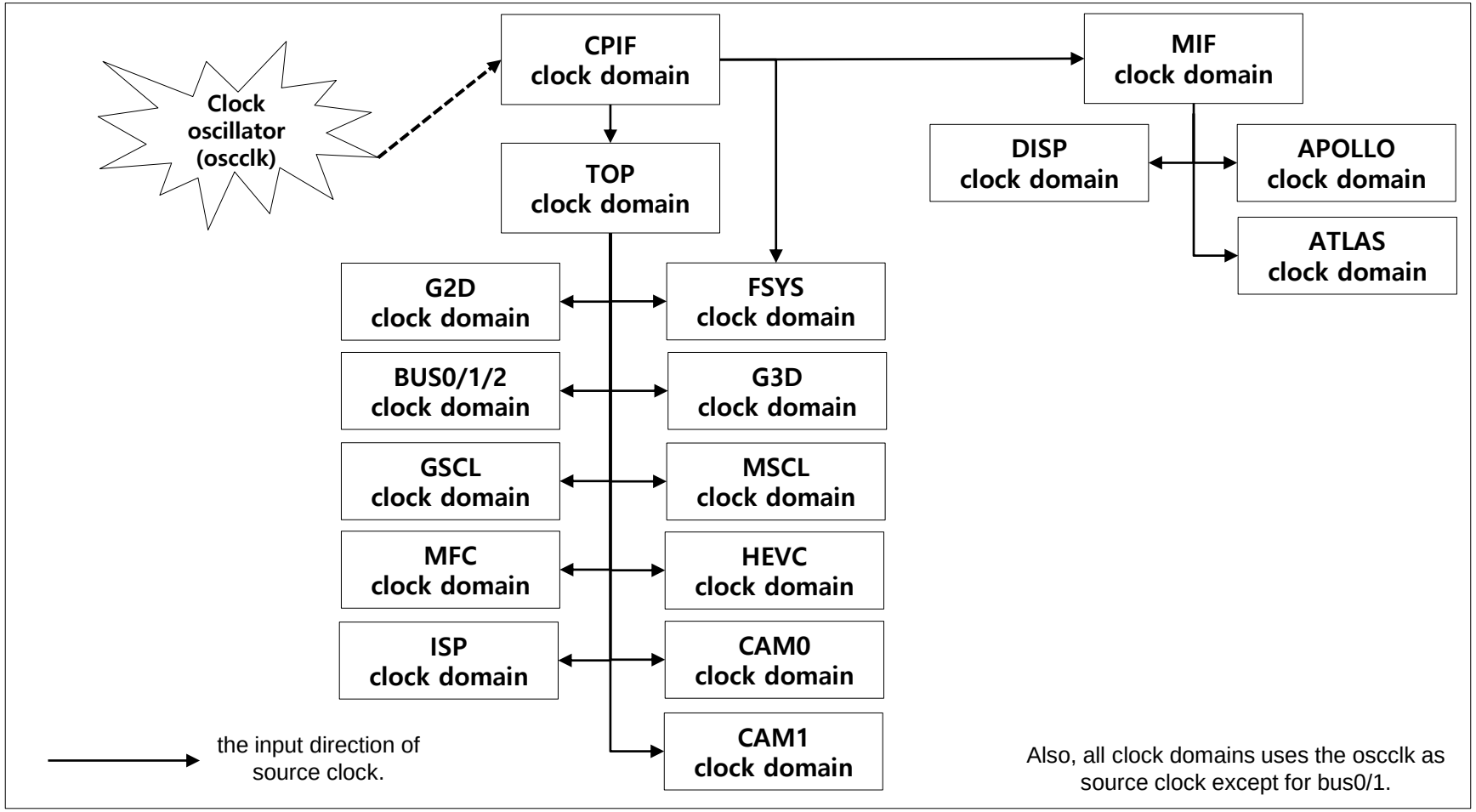


<Correlation between clock domain and power domain about display module in Exynos5433 SoC>

- But, CCF does not support the runtime PM. The clock controller is not able to make the dependency with power domain.

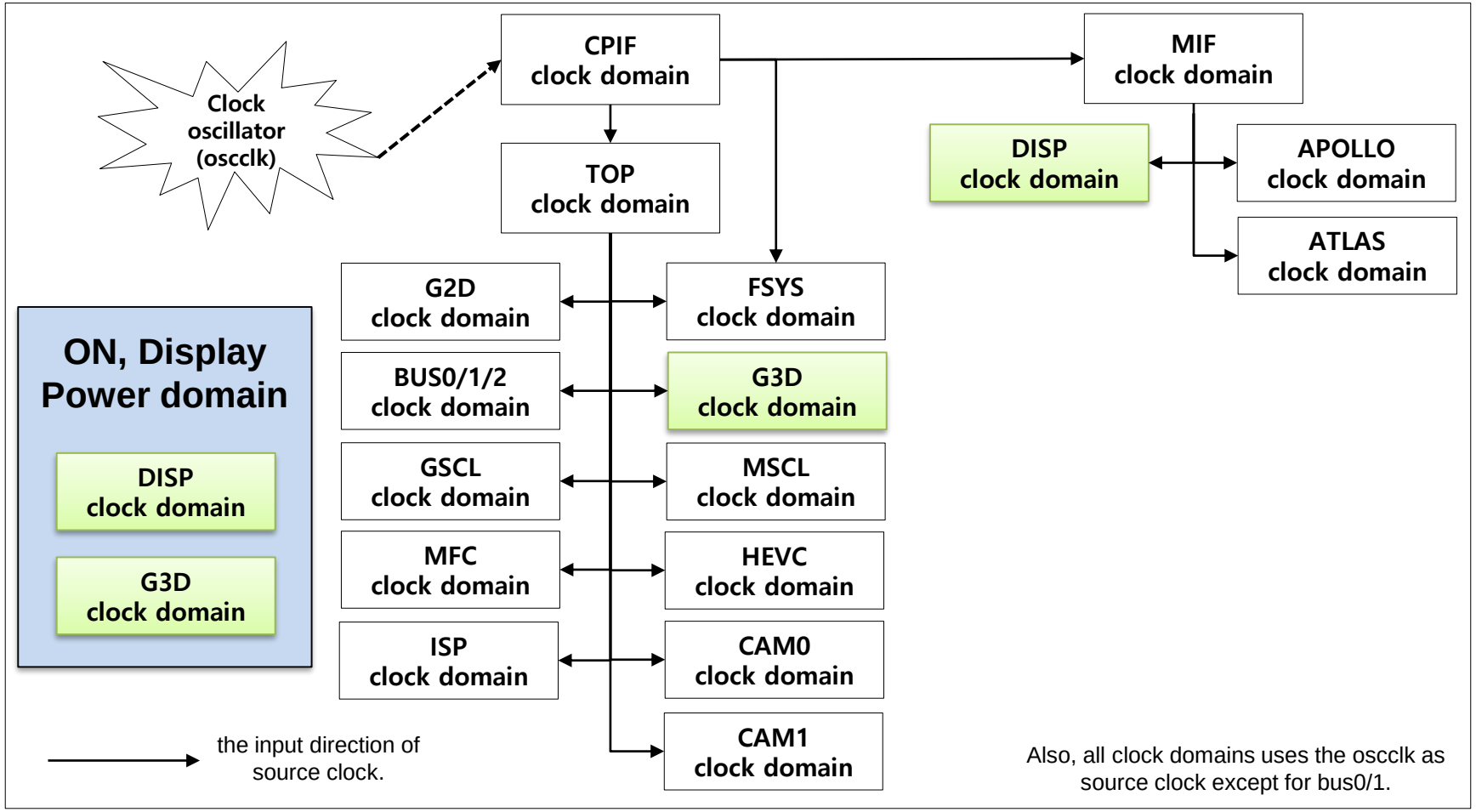
Dependency between power domain and CCF (3/6)

- Clock relationship in Exynos5433



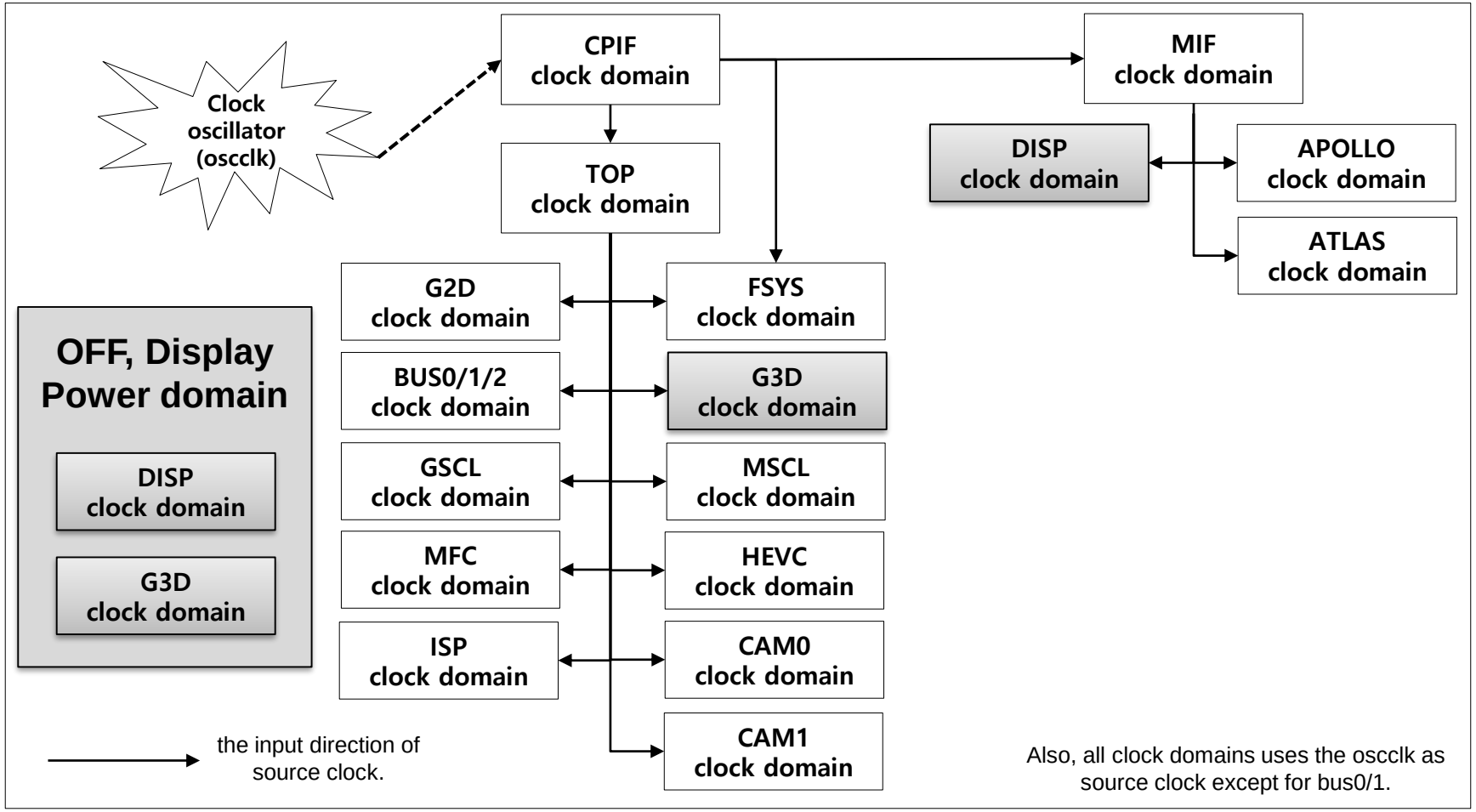
Dependency between power domain and CCF (4/6)

- Clock relationship in Exynos5433



Dependency between power domain and CCF (5/6)

- Clock relationship in Exynos5433



Dependency between power domain and CCF (6/6)

- How to fix the issue about dependency
  - CCF should support the runtime PM interface for the power domain.
  - The patches were posted to support the runtime PM for clock controller.
    - [PATCH v2 0/5] Add runtime PM support for clocks (on Exynos SoC example)
      - http://www.spinics.net/lists/arm-kernel/msg532798.html

Decompression of ARM 64bit kernel image

- ARM 64bit kernel does not support the decompression of kernel image.

- Using decompression feature of u-boot bootloader for compressed kernel image if you use the compressed kernel image.

More issues

TODO for TM2

- Exynos5433 Device-Tree
- Exynos5433-TM2/TM2E Device-Tree
- Support the multiple IORESOURCE_MEM for Pinctrl
- Peripheral devices for TM2
  - Touchscreen/key, Fuel-gauge/Charger, Sensorhub and so on.

Demo

Thank You!

References (cont.)

- Tizen
  - http://www.tizen.org
- ARMv8-A Reference Manual
  - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.architecture/index.html
- ARMv8-A Architecture
  - http://www.arm.com/products/processors/armv8-architecture.php
- KASAN
  - https://www.kernel.org/doc/Documentation/kasan.txt
- UBSAN
  - https://www.kernel.org/doc/Documentation/ubsan.txt
- PSCI (Power State Coordinate Interface)
  - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.den0022c/index.html
- HMP git repository by linaro
  - http://git.linaro.org/?p=arm/big.LITTLE/mp.git
- IKS (In Kernel Switcher)
  - https://events.linuxfoundation.org/images/stories/slides/elc2013_poirier.pdf
- ARM Generic Timer (cont.)
  - https://www.kernel.org/doc/Documentation/devicetree/bindings/arm/arch_timer.txt
  - http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.den0024a/BABGBFBF.html

References

- ARM Generic Timer
  - commit 65b5732d241b8 (“clocksource: arch_timer: Allow the device tree to specify uninitialized timer registers”)
  - commit d6ad36913083d (“clocksource: arch_timer: Only use the virtual counter (CNTVCT) on arm64”)
- GCC option ‘-O2’
  - https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html
- ‘FMOV’ assembly command
  - http://www.keil.com/support/man/docs/armclang_asm/armclang_asm_pge1427897629393.htm
- Fix ABBA deadlock patch between regmap and CCF
  - i2c: s3c2410: fix ABBA deadlock by keeping clock prepared
    - https://patchwork.kernel.org/patch/5659351/
  - i2c: exynos5: Fix possible ABBA deadlock by keeping I2C clock prepared
    - https://patchwork.kernel.org/patch/8859551/
  - [RFC 00/17] clk: Add per-controller locks to fix deadlocks
    - https://lkml.org/lkml/2016/8/16/442